

Project: Kinematics Pick & Place

Writeup Template: You can use this file as a template for your writeup if you want to submit it as a markdown file, but feel free to use some other method and submit a pdf if you prefer.

****Steps to complete the project:****

1. Set up your ROS Workspace.
2. Download or clone the [project repository](<https://github.com/udacity/RoboND-Kinematics-Project>) into the **src** directory of your ROS Workspace.
3. Experiment with the forward_kinematics environment and get familiar with the robot.
4. Launch in [demo mode](<https://classroom.udacity.com/nanodegrees/nd209/parts/7b2fd2d7-e181-401e-977a-6158c77bf816/modules/8855de3f-2897-46c3-a805-628b5ecf045b/lessons/91d017b1-4493-4522-ad52-04a74a01094c/concepts/ae64bb91-e8c4-44c9-adbe-798e8f688193>).
5. Perform Kinematic Analysis for the robot following the [project rubric](<https://review.udacity.com/#!/rubrics/972/view>).
6. Fill in the `IK_server.py` with your Inverse Kinematics code.

[//]: # (Image References)

[image1]: ./misc_images/misc1.png
[image2]: ./misc_images/misc2.png
[image3]: ./misc_images/misc3.png
[image4]: ./misc_images/misc4.png

[Rubric](<https://review.udacity.com/#!/rubrics/972/view>) Points

Here I will consider the rubric points individually and describe how I addressed each point in my implementation.

Writeup / README

1. Provide a Writeup / README that includes all the rubric points and how you addressed each one. You can submit your writeup as markdown or pdf.

First of all symbols are defined for all the params that's going to be used in the DH table. Then use the DH params to define DH table with correct values. Get the correct values of α, a, d

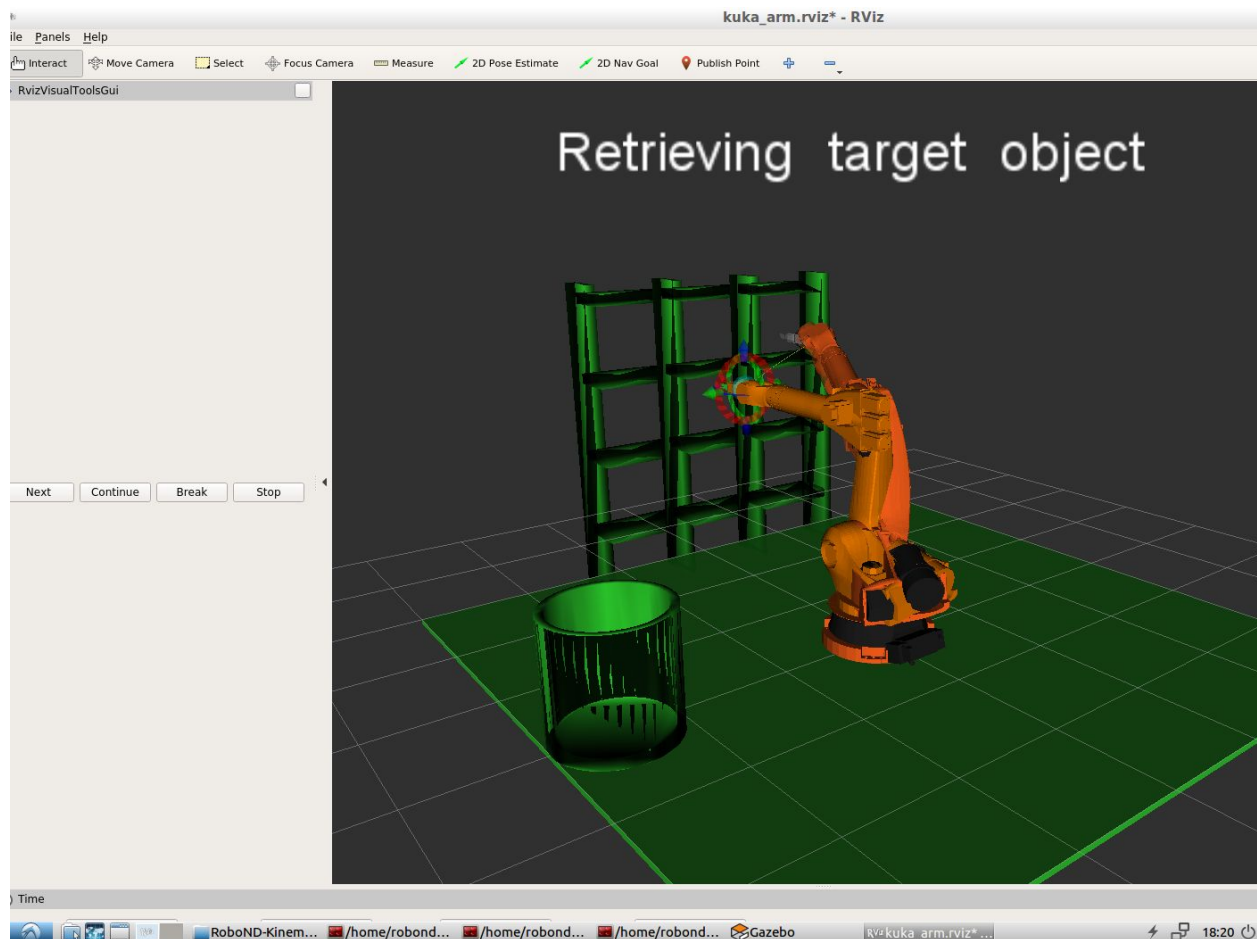
and theta using kr210 urdf file. From there onwards for every pose received , extract the end effector position and orientation. Do the correction to end effector position to be align with DH frame of reference. Use this information to get position of wrist center. Once the position of WC is known, use trigonometry to find theta1, theta2 and theta3. Once all three angles are known use this to find out transformation from base link to link 3. Once we know the transformation from T0_3 use it obtain transformatin from WC to endEffector. Once transformation from WC to end effector is known we can use different formulas used in class to find rest of theta4, theta5 and theta6 angles and thus we can solve inverse kinematics.

Kinematic Analysis

1. Run the forward_kinematics demo and evaluate the kr210.urdf.xacro file to perform kinematic analysis of Kuka KR210 robot and derive its DH parameters.

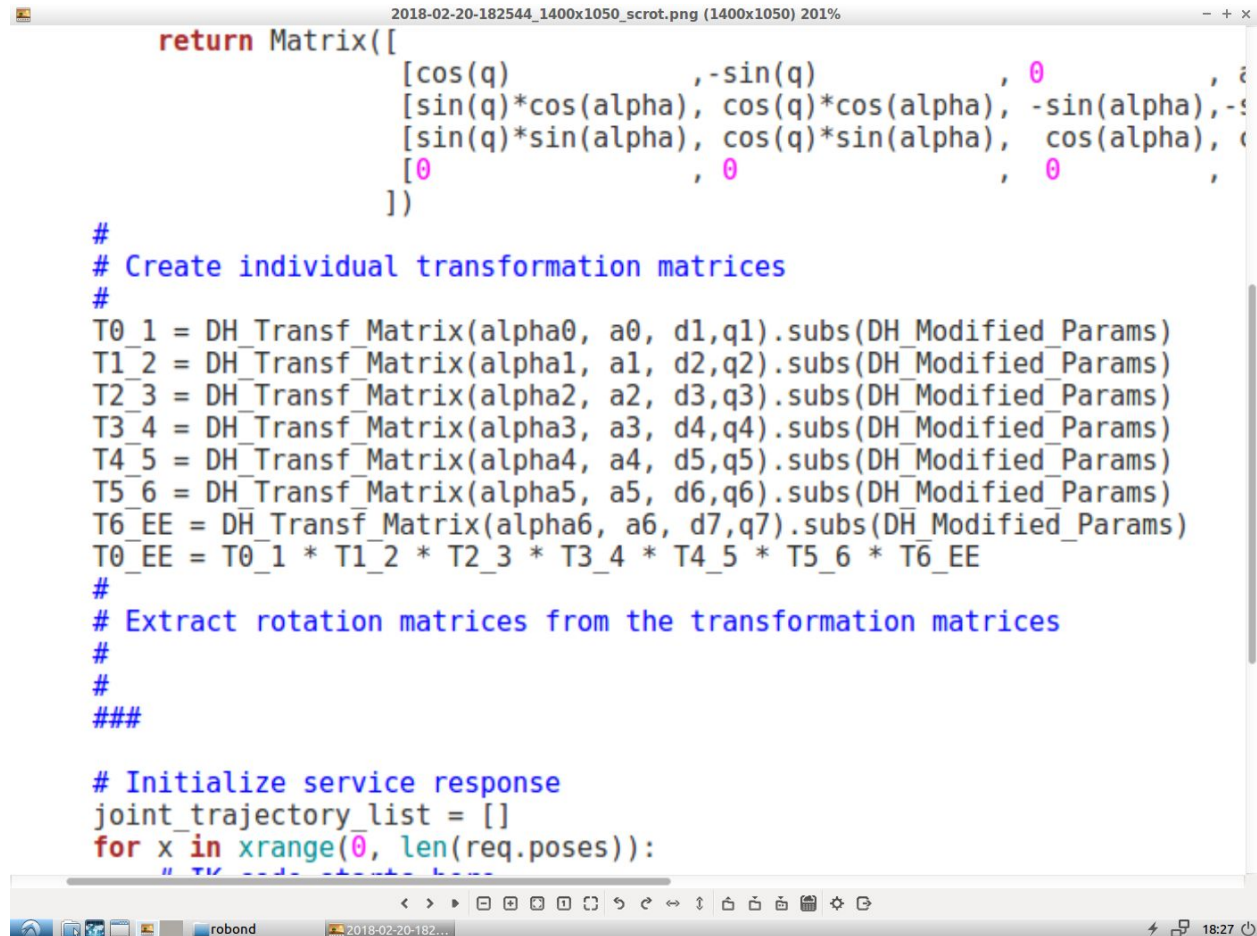
Here is an example of how to include an image in your writeup.

![alt text][image1]



2. Using the DH parameter table you derived earlier, create individual transformation matrices about each joint. In addition, also generate a generalized homogeneous transform between base_link and gripper_link using only end-effector(gripper) pose.

![alt text][image3]



```

return Matrix([
    [cos(q), -sin(q), 0, 0],
    [sin(q)*cos(alpha), cos(q)*cos(alpha), -sin(alpha), 0],
    [sin(q)*sin(alpha), cos(q)*sin(alpha), cos(alpha), 0],
    [0, 0, 0, 1]
])

#
# Create individual transformation matrices
#
T0_1 = DH_Transf_Matrix(alpha0, a0, d1,q1).subs(DH_Modified_Params)
T1_2 = DH_Transf_Matrix(alpha1, a1, d2,q2).subs(DH_Modified_Params)
T2_3 = DH_Transf_Matrix(alpha2, a2, d3,q3).subs(DH_Modified_Params)
T3_4 = DH_Transf_Matrix(alpha3, a3, d4,q4).subs(DH_Modified_Params)
T4_5 = DH_Transf_Matrix(alpha4, a4, d5,q5).subs(DH_Modified_Params)
T5_6 = DH_Transf_Matrix(alpha5, a5, d6,q6).subs(DH_Modified_Params)
T6_EE = DH_Transf_Matrix(alpha6, a6, d7,q7).subs(DH_Modified_Params)
T0_EE = T0_1 * T1_2 * T2_3 * T3_4 * T4_5 * T5_6 * T6_EE
#
# Extract rotation matrices from the transformation matrices
#
#
###

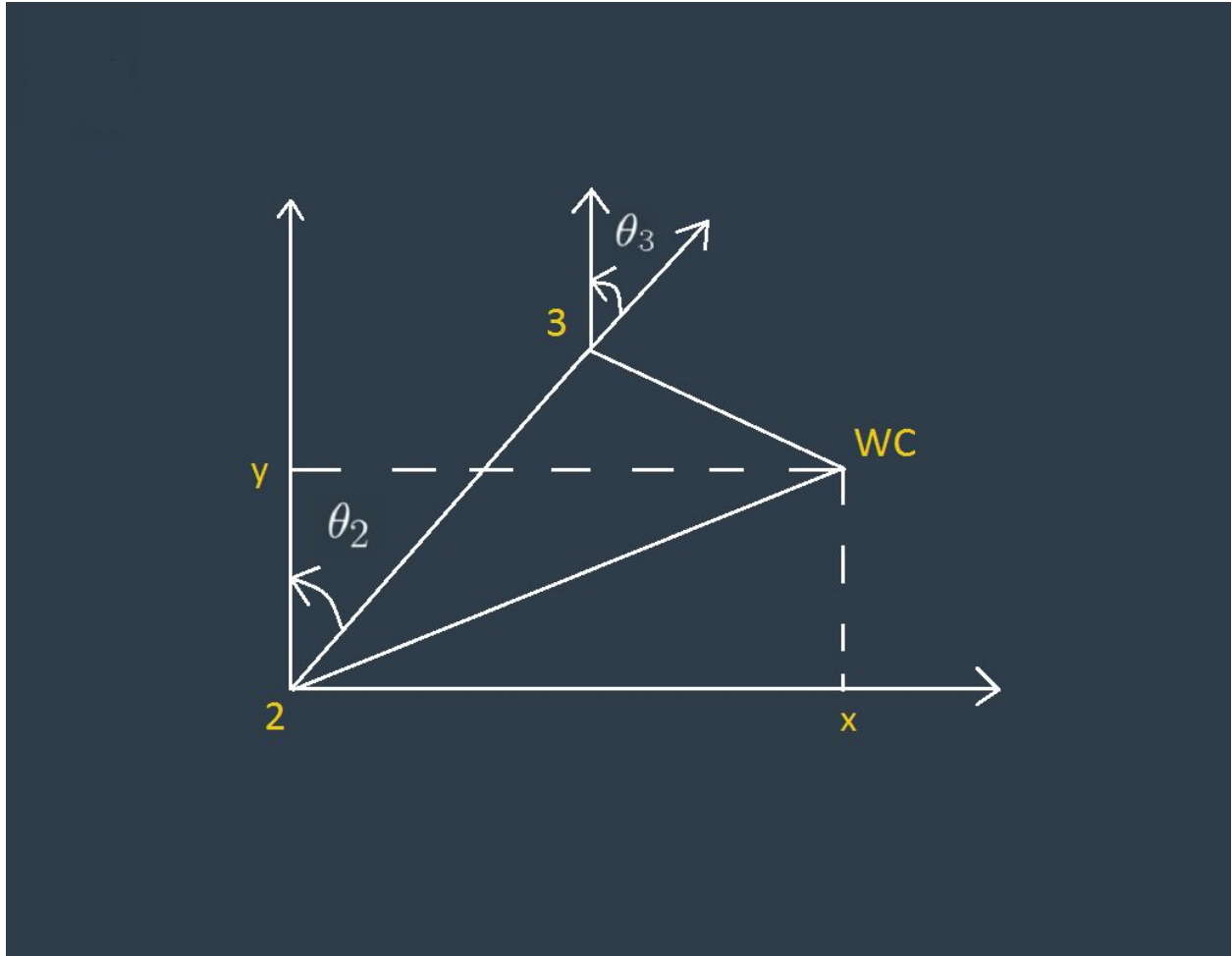
# Initialize service response
joint_trajectory_list = []
for x in xrange(0, len(req.poses)):
    " TK code starts here

```

3. Decouple Inverse Kinematics problem into Inverse Position Kinematics and inverse Orientation Kinematics; doing so derive the equations to calculate all individual joint angles.

And here's where you can draw out and show your math for the derivation of your theta angles.

![alt text][image4]



Project Implementation

1. Fill in the `IK_server.py` file with properly commented python code for calculating Inverse Kinematics based on previously performed Kinematic Analysis. Your code must guide the robot to successfully complete 8/10 pick and place cycles. Briefly discuss the code you implemented and your results.

Given more time I would like to work on making my code more quicker to reach the destination points.

And just for fun, another example image:

![alt text][image2]

